applets x, y and z pointing to a data file **706**, which contains a directory of tasks with identifiers to link them to one or more applets (for simplification, only three tasks are illustrated in data file **706**). It should be noted that it is within the scope of the present invention that the data file be instead implemented from within the file system of the operating system or as an embedded resource. All similar or equivalent implementations are also within the scope of the present invention.

[**0057**] As will be recognized by those skilled in the art, the specific configuration illustrated in **FIG. 7** is only one of many ways to implement the described functionality. All similar variations are to be considered within the scope of the present invention. While **FIG. 7** would seem to imply that an applet must register a category before pointing to a data file, this is not necessarily the case. In accordance with one embodiment, an applet is configured to register these two pieces of data independently.

[**0058**] In accordance with one aspect of the present invention, for each task, the metadata in file **706** will include any of a display name (optionally localized into multiple languages), a task name, a descriptive tool tip (optionally localized), a command to run when the task is invoked, a task description, and optional states to check before displaying the task. Other descriptive metadata, such as keywords or related search queries, are also potentially included to enable the tasks to be exposed in task search results. An example of a state check is to not show an "Add New User" task if the current user is not an Administrator on the computer. In one embodiment, there is additional metadata that determines which tasks appear when an applet appears under different categories, as well as the order in which tasks appear. In one embodiment, task names and descriptions can be overridden when appearing under specified categories.

[**0059**] As has been described, one aspect of the present invention pertains to a tool provided to support an author in making the registry and/or data file entries necessary to manipulate the extensible framework. In one embodiment, as was alluded to in **FIG. 6**, a visual editor is configured to abstract the underlying data management schema for task authors by enabling them to information such as, but not necessarily limited to, a task name, a description, and a command. This saves the author the trouble of direct editing, such as direct editing of an XML file. The tool also illustratively supports a simplified ability (i.e., without direct coding) to specify task state checks. The tool also illustratively supports a simplified ability to specify which tasks appear under which applets for each category, as well as an ability to override task names and descriptions. As has been described, one embodiment pertains to the provision of a virtual screenshot demonstrating what the applet will look like with its list of tasks. Such a screenshot can be drawn so the user doesn't need to actually load the task in the control panel to see what is being edited.

[**0060**] In accordance with another embodiment, the described data management scheme includes an internal data file (e.g., an XML file) (not illustrated in **FIG. 7**) that references tasks that should appear under each category in the category view home page (e.g., **FIG. 2**). A similar file can be implemented to determine tasks displayed in the applet views (e.g., **FIGS. 3 & 4**). As categories or applets are added, their associated tasks can be designated as selectable hyperlinks. Those skilled in the art will appreciate that there are many obvious means for implementing a system to police or restrict which tasks are or are not displayed in various views or windows.

[**0061**] It should be noted that the embodiments of data management associated with the present invention are not limited to a registry system associated with an operating system. Other registry systems can be similarly implemented without departing from the scope of the present invention. Similarly, the embodiments of data management are not limited to implementation of any particular data file system. For example, any data file other than XML data files can be similarly implemented without departing from the scope of the present invention. Also, without departing from the scope of the present invention, the line between the functionality of the registry system and the functionality of the data file can be manipulated. For example, elements tracked in the data file can alternatively be tracked in the registry, and vice versa.

[**0062**] In summary, embodiments of the present invention pertain to a system that enables users to more quickly jump to the area of a control panel applet's user interface that relates to the desired task. Task buttons in the control panel category view are illustratively associated with metadata to support the task shortcut. Instead of requiring users to open an applet then search its windows or tabs for the task they intend to complete, they can simply click a task button that enables a jump straight to an appropriate applet component for performing the task. In one embodiment, the jump is straight to an active content wizard or guided help component that provides guidance for completing the task.

[**0063**] Other embodiments of the present invention pertain to an underlying XML file metadata schema and registration system that supports applet developers in the addition and manipulation of tasks. Through this schema, developers are not forced to write compiled code modules to enable their applets to show tasks in a category view; instead, a data file contains the task metadata. In one embodiment, a tool (e.g., a graphical, visual editor) is provided to support applet developers. In one embodiment, the tool enables applet developers to author tasks in a manner that is far easier than utilization of a text editor.

[**0064**] Although the present invention has been described with reference to particular embodiments, workers skilled in the art will recognize that changes may be made in form and detail without departing from the spirit and scope of the invention.

What is claimed is:

1. A computer-implemented method for enabling a user to efficiently navigate to a portion of a user interface configured to support performance of a particular task, the method comprising:

receiving from the user an input that includes a description of the particular task; and

returning to the user a result set that includes a plurality of candidate tasks that correspond to said description, the plurality of candidate tasks being displayed relative to an indication of at least one corresponding applet.

2. The method of claim 1, wherein returning further comprises returning a result set that includes a plurality of